

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

In re Application of: Suresh Singamsetty Kumar, et al.  
Serial No.: 09/851,721  
Filing Date: May 8, 2001  
Group Art Unit: 2194  
Examiner: Diem K. Cao  
Title: *FINITE STATE MACHINE IN A  
PORTABLE THREAD ENVIRONMENT*

**MAIL STOP AF**  
Commissioner for Patents  
PO Box 1450  
Alexandria, VA 22313-1450

Dear Sir:

**PRE-APPEAL BRIEF REQUEST FOR REVIEW**

The following Pre-Appeal Brief Request for Review ("Request") is being filed in accordance with the provisions set forth in the Official Gazette Notice of July 12, 2005 ("OG Notice"). Pursuant to the OG Notice, this Request is being filed concurrently with a Notice of Appeal. The Applicants respectfully request reconsideration of the rejection of all claims in the Application.

**REMARKS**

In the prosecution of the present Application, the Examiner's rejections and assertions contain clear errors of law. Most notable of the legal errors is a failure of the Final Office Action to establish a *prima facie* rejection of the claims in the application under 35 U.S.C. § 103. The Final Office Action rejected Claims 1-3 under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 6,430,593 issued to Lindsley ("*Lindsley*") in view of Pai, et al., "Flash: An Efficient and Portable Web Server" (hereafter "*Pai*"). Claims 4-20 were rejected under 35 U.S.C. § 103(a) as being unpatentable over *Lindsley* in view of *Pai* and further in view of U.S. Patent No. 5,727,214 issued to Allen (hereafter "*Allen*"). These rejections fail to meet the required *prima facie* standard for obviousness for at least the two reasons set forth below.

First, the combination of *Lindsley* and *Allen* is improper because the proposed combination would render *Lindsley* unsatisfactory for its intended purpose. If a "proposed modification would render the prior invention being modified unsatisfactory for its intended purpose, then there is not suggestion or motivation to make the proposed modification." MPEP § 2143.01. *Lindsley* discloses a system that processes a plurality of tasks by using a task scheduling accelerator. (*Lindsley*; col. 5, ll. 59-67; col. 6, ll. 1-4; Abstract). An intended purpose of *Lindsley* is to "compute schedule decisions in parallel with activity on the host processor." (*Lindsley*; col. 5, ll. 59-67; col. 6, ll. 1-4; Abstract). In contrast, *Allen* discloses a matrix configured for processing an event as a *single* threaded object. The matrix in *Allen* is configured for handling "only one thread of execution at a time." (*Allen*; col. 8, ll. 2-5; col. 12, ll. 7-11). Modifying *Lindsley* to map information to the matrix in *Allen* would require *Lindsley* to process events "one thread of execution at a time." (*Allen*; col. 8, ll. 2-5). Such a modification would render *Lindsley* unsatisfactory for its intended purpose of "computing scheduling decisions *in parallel* with activity on the host processor." (*Lindsley*; col. 5, ll. 59-67; col. 6, ll. 1-4) (emphasis added). Because the proposed modification would render *Lindsley* unsatisfactory for its intended purpose, there is no motivation to combine *Lindsley* and *Allen*. Thus, the proposed combination is improper. Accordingly, Applicants respectfully request the withdrawal of the proposed combination.

The Examiner has not overcome this argument. In response to the above argument, the Examiner merely stated:

[T]he task scheduling accelerator is used to compute schedule decision for plurality of task, and it is not the same as multiple threads are executed at the same time. It is well known in the art that even if there is multiple threads are existed in the system, only one thread is executed by the operating system at a time.

(Office Action dated 11/29/2005; page 6). The Examiner misses the point. Even if the task scheduling accelerator in *Lindsley* is not the same as multiple threads, *Lindsley* clearly requires that the system “compute schedule decisions in parallel with activity on the host processor.” (*Lindsley*; col. 5, ll. 59-67; col. 6, ll. 1-4; Abstract) (emphasis added). Because the matrix in *Allen* is configured for single threaded objects -- that is, for handling “only one thread of execution at a time” -- the proposed combination results in a system where decisions are process *serially* rather than *in parallel*. (*Allen*; col. 8, ll. 2-5; col. 12, ll. 7-11). Such a modification would plainly render *Lindsley* unsatisfactory for its stated purpose. Thus, the proposed combination must be withdrawn.

Second, clear errors of law underlie the Examiner’s conclusion that the *Lindsley-Pai* combination teaches that “a plurality of threads send PTE messages to each other while cooperatively completing a task” as recited in Claim 1. The Final Office Action acknowledges that *Lindsley* does not disclose this element of Claim 1. (Final Office Action; page 2). Rather, the Examiner relies on *Pai* for this element. However, the Examiner failed to clearly identify which component of *Pai* was being equated with the above element of Claim 1 until the Advisory Action dated April 20, 2006. Significantly, *Pai* discloses four different architectures for a web server: multi-process (MP), multi-thread (MT), single-process event-driven (SPED), and asymmetric multi-process event-driven (AMPED). (*Pai*; § 3). The Office Action dated January 11, 2005 clearly relies on the MT server for the above element of Claim 1. (Page 3 (citing § 3.2, ¶ 2 of *Pai*)). Subsequently, however, the Office Action dated November 29, 2005 seems to rely on the AMPED server for the above element of Claim 1. (Page 3 (citing §§ 3.4 and 5.1 of *Pai*)). Thus, it was not clear whether the Examiner was relying on the MT architecture or the AMPED architecture until the Advisory Action dated April 20, 2006, wherein the Examiner clarified that the AMPED server and not the MT server was being equated with the above element of Claim 1. (Advisory Action dated 4/20/2006; page 2). It is impermissible to wait until after a final rejection to clearly identify the particular part of the reference relied upon. “When a reference is complex or shows or describes inventions other than that claimed by the applicant, the particular part relied on

must be designated as nearly as practicable.” 37 CFR § 1.104(c)(2). Because the component being relied upon was not clearly identified until after the final rejection, the rejection of Claim 1 is improper.

Furthermore, the portion of *Pai* relied upon by the Examiner -- the AMPED server -- falls far short of teaching, suggesting, or disclosing that “a plurality of threads send PTE messages to each other while cooperatively completing a task” as recited in Claim 1. The AMPED architecture uses a server process assisted by helper processes. (*Pai*; § 5.1). “The helper processes are responsible for performing all of the actions that may result in synchronous disk activity.” (*Pai*; § 5.1). Each helper process “operates synchronously, waiting on the server for new requests and handling only one request at a time.” (*Pai*; § 5.1). Most notably, *Pai* specifically states: “[t]o minimize interprocess communication, helpers *only* return a completion notification *to the server*, rather than sending any file content they may have loaded from the disk.” (*Pai*; § 5.1). Thus, the only communication disclosed in *Pai* is between a helper and the server. There is nothing in *Pai* that teaches, suggests, or discloses that “threads send PTE messages *to each other*” as recited in Claim 1. (Emphasis added). Furthermore, the communication between the helper process and the server in *Pai* occurs only *after* the completion of a request (it is “a *completion* notification”) -- not “*while* cooperatively completing a task” as recited in Claim 1. (Emphasis added). Thus, the *Lindsley-Pai* combination fails to teach, suggest, or disclose that “a plurality of threads send PTE messages to each other while cooperatively completing a task” as recited in Claim 1. As a result, the rejection is improper.

The Examiner has not overcome this argument. In the Advisory Action, the Examiner merely stated:

In the system AMPED, *Pai* teaches multiple threads are cooperative to complete a task (the main server process and multiple helper processes or thread; section 3.4 and section 5.1). Although *Pai* also discuss “minimize interprocess communication,” the interprocess communication is still occurred (section 5.1).

(Advisory Action dated 4/20/2006; page 2). The Examiner’s response falls short. Clearly, the phrase “minimize interprocess communication” in *Pai* does not teach, suggest, or disclose that “threads send PTE messages to each other while cooperatively completing a task” as recited in Claim 1. *Pai* describes “interprocess communication” by stating that “helpers *only* return a completion notification *to the server*, rather than sending any file content they may

have loaded from the disk.” (*Pai*; § 5.1). Thus, the only communication disclosed in *Pai* is between a helper and the server -- there are no threads sending messages “to each other” as recited in Claim 1. Furthermore, the communications in *Pai* are merely “completion” notifications -- the communications do not occur between threads “while cooperatively completing a task” as recited in Claim 1. (Emphasis added). Thus, the *Lindsley-Pai* combination fails to teach, suggest, or disclose that “a plurality of threads send PTE messages to each other while cooperatively completing a task” as recited in Claim 1.

### **CONCLUSION**

As a *prima facie* rejection has not been established against Applicants’ claims, Applicants respectfully request a finding of allowance of all claims in the Application.

To the extent necessary, the Commissioner is hereby authorized to charge any fees or credit any overpayments to Deposit Account No. 02-0384 of BAKER BOTTS L.L.P.

Respectfully submitted,

BAKER BOTTS L.L.P.  
Attorneys for Applicants



Samir A. Bhavsar  
Reg. No. 41,617

Date: May 26, 2006

### **CORRESPONDENCE ADDRESS:**

at Customer No.

**05073**